What is Wrong with Digital Documents? A Conceptual Model for Structural Cross-Media Content Composition and Reuse

Beat Signer

Vrije Universiteit Brussel
Pleinlaan 2, 1050 Brussels, Belgium
bsigner@vub.ac.be

Abstract. Many of today's digital document formats are strongly based on a digital emulation of printed media. While such a paper simulation might be appropriate for the visualisation of certain digital content, it is generally not the most effective solution for digitally managing and storing information. The oversimplistic modelling of digital documents as monolithic blocks of linear content, with a lack of structural semantics, does not pay attention to some of the superior features that digital media offers in comparison to traditional paper documents. For example, existing digital document formats adopt the limitations of paper documents by unnecessarily replicating content via copy and paste operations, instead of digitally embedding and reusing parts of digital documents via structural references. We introduce a conceptual model for structural cross-media content composition and highlight how the proposed solution not only enables the reuse of content via structural relationships, but also supports dynamic and context-dependent document adaptation, structural content annotations as well as the integration of arbitrary non-textual media types. We further discuss solutions for the fluid navigation and cross-media content publishing based on the proposed structural cross-media content model.

1 Introduction

In his 1945 seminal article 'As We May Think' [1], the visionary Vannevar Bush introduced the concept of the Memex, a prototypical hypertext machine for storing and accessing information on microfilm. As a knowledge worker, Bush was not happy with the current way of accessing information based on hierarchical classifications such as the Dewey Decimal Classification (DDC). As described in his article, the Memex was meant to enhance information management by introducing a superimposed metadata structure to be considered as a natural extension of human mind based on cross-references between different microfilms:

> *When data of any sort are placed in storage, they are filed alphabetically or numerically, and information is found (when it is) by tracing it down from subclass to subclass. [. . .] The human mind does not work that way. It operates by association.*

While Bush's vision is often accredited as being the origin of hypermedia systems, some of the early hypermedia pioneers, including Douglas Engelbart and Ted Nelson, brought the idea of defining associations between pieces of information into digital age. The Memex offered only limited structural representation of information since it was based on printed documents (i.e. microfilm document pages). The only available document structure was the concept of a page which implied that links (associations) could only be defined at the page level. In his Xanadu document model [2], Ted Nelson introduced the idea of so-called deep documents, where snippets of information can be reused in higher-level document structures via the concept of *transclusion*. The Xanadu document model no longer requires textual information to be replicated since the same content can be embedded in different documents via structural references (transclusions) in combination with a versioning mechanism.

In the mid 70's, researchers at Xerox PARC coined the term 'What you see is what you get' (WYSIWYG) which would become a de facto standard for document representations including the Portable Document Format (PDF), the Word Document Format and most other existing document formats. Unfortunately, the WYSIWYG representation did not take into consideration richer digital document representations such as the one proposed in Engelbart's oN-Line System (NLS) [3] or Nelson's Xanadu project. It further did not address the new possibilities that digital document formats could offer in comparison to printed media and the computer rather became degraded to a paper simulator as criticised by Nelson [4]:

> *Most people don't understand the logic of the concept: "What You See Is What You Get" is based on printing the document out ("get" means "get WHEN YOU PRINT IT OUT"). And that means a metaphysical shift: a document can only consist of what can be printed! This re-froze the computer document into a closed rectangular object which cannot be penetrated by outside markings (curtailing what you could do with paper). No marginal notes, no sticky notes, no crossouts, no insertions, no overlays, no highlighting—PAPER UNDER GLASS.*

Unfortunately, these limitations in terms of missing document structure in combination with hierarchical file systems led to unsatisfactory information management solutions. Often users have to replicate content—for example if the same picture is going to be used in multiple documents—due to monolithic and closed document formats. Driven by recent developments on the Web where document structures are made explicitly available, for example through the increasing use of the Extensible Markup Language (XML), people start to realise that also desktop applications could profit from richer hypermedia structures to overcome some of the limitations of hierarchical file systems.

We strongly believe that existing desktop applications can provide enhanced functionality based on a richer digital document representation that pays attention to structural semantics as well as bidirectional inter-document relationships. This functionality should not be provided in a proprietary and application-specific manner on top of existing file systems, but rather form part of the core

functionality offered by an enhanced file system. In this paper, we introduce a general and media-neutral structural cross-media document model based on a core link metamodel which can be extended via resource plug-ins to support existing as well as emerging document formats. We further show that the application of such a cross-media document layer may lead to enhanced information management functionality and ultimately result in a remediation of existing "paper simulation" approaches.

We begin in Sect. 2 by providing an overview of existing digital document representations that are used on local file systems as well as on the Web. We discuss some advantages of existing document formats and highlight potential future improvements for dealing with cross-media information management. In Sect. 3 we introduce the main concepts of our conceptual model for structural cross-media content composition and discuss some differences to existing document formats. We then present an implementation of the model based on an object database management solution and outline potential authoring and publishing components. Various application scenarios of the presented cross-media document model, including a prototype of an associative file system, are highlighted in Sect. 5. Concluding remarks are given in Sect. 6.

2 Background

As denoted in the previous section, early hypermedia and document models offered rich features for addressing parts of documents, defining bidirectional relationships between documents as well as parts of documents and the structural embedding of documents or parts of documents via transclusion. However, the WYSIWYG concept offered by early desktop solutions sacrificed the rich digital document models in favour of simpler implementations and visualisations of documents and led to some of the document formats that are still in use today.

In most existing applications, a document is represented as a single file that is stored somewhere in the file system. Furthermore, the content of a document is normally stored in a proprietary format which makes it hardly impossible for third-party applications to access any structural metadata that is completely controlled by the monolithic application "owning" the document. These closed document formats prevent third-party applications to offer supplemental services related to parts of a document and make the external annotation of document substructures impossible.

There are two types of structural metadata that have to be considered: the *within document structure* describing different parts of a document (e.g. sections, images, etc.) and the *external document structure* which superimposes a structure on multiple documents (e.g. to organise projects with multiple documents). The within document structure is normally fixed and only accessible to the application that has been used to create the document, whereas an explicit external document structure is not offered by most existing file systems. A problem of proprietary document formats is not only that it is difficult to access the corresponding data from within third-party applications, but also

the document rendering functionality is generally not portable between different document types. This results in major development efforts for the visualisation of particular document formats and limits innovation since only large companies have the necessary resources to develop advanced document renderers. To overcome this problem, Phelps and Wilensky [5] proposed an architecture that separates the document format from its associated functionality represented by reusable and composable behaviour objects.

The paper simulating document formats (e.g. PDF or Word Document Format) support only a single WYSIWYG document structure. However, in a digital environment that offers many different non-printable media types, this kind of representation might not be appropriate for certain mixed-media information compositions. A flexible solution should rather support some basic structuring mechanism and allow for different structural views on top of existing pieces of information. Note that such a basic structuring mechanism should be general enough to support existing as well as emerging document formats as we show when introducing our structural cross-media model in the next section. Of course, the WYSIWYG representation might always be one of the potential structural representations, but it should not be the only possible one.

As mentioned earlier, an external document structure to define the semantics of a set of documents is not supported by most file systems. The general way to organise documents of a project is to make use of hierarchical folder structures. However, the hierarchical organisation of documents has a number of limitations. In most file systems a file can only be in a single folder since also folders are inspired by their physical counterparts. The affordances of objects in the physical space have again been copied to digital space rather than making use of the richer functionality that could be offered for digital documents in terms of *multiple classification*. Without any physical constraints, there is no reason why a file should not be put into multiple folders. Nevertheless, the lack of this functionality leads to an unnecessary replication of information since the same file is copied to different projects folders. Even worse, there is no relation between multiple instances of the same file which makes it possible that different inconsistent versions of the same document might exist. Last but not least, the underlying file system as well as the different applications have no understanding of the semantics that a user encodes in these folder structures and therefore cannot profit from exploiting this structural semantics. Alternative non-hierarchical access forms to traditional file systems are also investigated by new collaborative tangible user interfaces [6]. When we introduce potential applications of our structural cross-media content model in Sect. 5, we will list some applications that offer enhanced information management functionality by making use of the underlying non-hierarchical structural metadata.

In the early days, the Web had similar limitations in terms of not providing access to document structures. The Hypertext Markup Language (HTML) is a mix of content and visualisation that interweaves content with structural and navigational elements. While earlier hypertext models envisioned a separation of content and external link structures, the Web mixes these concepts. HTML only

offers embedded and unidirectional links, which means that only the owner of a document can add links to it and links can not be traced back. HTML offers a very restricted form of transclusion for embedded images [7]. An external image can be embedded in an HTML document by referencing its URL via the `src` attribute of an `<img>` tag. However, we can only embed the entire image and it is not possible to address parts of an image. While this limited form of transclusion is supported for images, the structural composition is not possible for textual HTML content. More recently, it has been realised that the external linking and annotation of webpages offers an immense flexibility in terms of dynamic and adaptive content composition. The hypermedia structures that are used on the Web, start to blur the notion of classical document boundaries since it is no longer clear which hyperlinked resources should be counted as part of a particular document.

The introduction of the Extensible Markup Language (XML) was an important step to open the structure of documents. In combination with the XPointer and XPath languages for addressing parts of an XML structure, the XML Linking Language (XLink) [8] can be used to define external multi-directional links. The use of XML as a general representation for cross-media documents might be problematic since XML itself puts some constraints on the structural grouping of elements by offering a hierarchical document model.

Based on the opening of web document formats, the Annotea project [9] introduced a solution for external collaborative semantic annotations that is, for example, used in the W3C's Amaya[1] web browser to edit third-party webpages. More recently, the use of mashups, another form of transclusion where snippets of content and services are composed to form new content and services, becomes more important on the Web. Another form of transclusion is provided by CrystalBox[2], an agile software documentation tool that embeds code as well as other resources into HTML pages and other web documents by simply defining references to the corresponding resources in an online software repository.

In the near future, we can see a trend to apply ideas from the Web in desktop applications, thereby reducing the gap between applications running on a local machine and services or documents available on the Web. Some companies already started to open their proprietary document formats. For example, the latest Microsoft Office suite uses the Office Open XML [10] standard to represent text documents, presentations as well as other documents. This interesting development makes the structural document components (e.g. a slide of a PowerPoint presentation) accessible to third-party applications and tools which can reuse parts of a document based on structural references (e.g. via an XPointer expression to an Office Open XML document). While this solution uses XML as a tool for providing access to the application-specific structural semantics, we propose a set of structural concepts to be offered at the file system level. This has the advantage, that any application will be able to make use of the additional metadata.

[1] http://www.w3.org/Amaya/
[2] http://crystalbox.org

Existing hypermedia solutions have been challenged by more recent approaches such as structural computing where structure is treated as a first-class citizen and the focus is no longer on the data [11]. As we show in the next section, our model treats data, navigational as well as structural metadata at the same level. Note that also new hypermedia data structures such as Zzstructures, polyarchies and mSpaces might be used for organising documents [12].

The idea of open document formats has been addressed by Rivera et al. when discussing future directions of their OMS-FS object file system [13]. They recommended to represent documents as containers of different media components that are associated with the corresponding digital functionality (e.g. editor, viewer etc.). This approach is somehow related to the visualisation that we later introduce for our structural cross-media document model.

It is likely that in the future we will see a reconciliation of digital document spaces with tangible objects such as interactive paper documents [14], RFID-tagged objects and other physical resources. A general structural cross-media document model should therefore be able to define structural compositions of mixed digital and physical resources. In the next section, we introduce our general conceptual model for structural cross-media content composition and explain how this core model can be used and extended to address some of the issues mentioned above.

3 Structural Cross-Media Model

In this section we present our structural cross-media document model and explain how its minimal set of concepts can be extended via resource plug-ins to support different types of documents. The presented structural model is an application of our general resource-selector-link (RSL) model [15] that was originally developed for the cross-media linking of digital and physical resources. However, we will show that the structural composition of different resources can be seen as a specialisation of the general link concept. The core components of the RSL model with its structural link support are shown in Fig. 1.

Our structural cross-media composition model has been defined using the OM data model [16] which integrates concepts from both entity relationship (ER) and object-oriented data models. The OM model distinguishes between *typing* and *classification*. The typing deals with the representation of entities by objects with their attributes and methods. On the other hand, the semantic roles of individual object instances are managed by classification via named collections which are represented by the rectangular shapes in Fig. 1. Furthermore, the OM model offers a first-class association construct which is represented by the oval shapes with theirs corresponding cardinality constraints. Note that associations might be ordered and that a ranking over an association is highlighted by putting the name of the association between two vertical lines (e.g. |`HasChild`|). It is out of the scope of this paper to provide a full overview of the OM model but a detailed description can be found in [16].

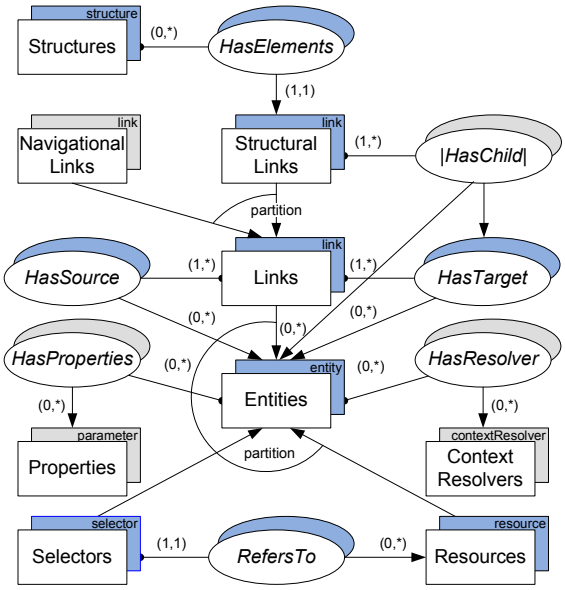

Fig. 1. Structural cross-media composition model

The most general concept introduced by our RSL model is the notion of an `entity` type. All entity instances are further grouped by the corresponding `Entities` collection. Our conceptual cross-media link model further introduces three concrete specialisations of the abstract entity concept as indicated by the corresponding `resource`, `selector` and `link` subtypes.

The `resource` type represents any particular resource can be used as element in a composition. A resource itself is an abstract concept and for each specific media type, the corresponding resource plug-in has to be provided. A resource plug-in stores additional metadata about the corresponding media type (e.g. a URL attribute for the HTML resource plug-in).

Links between different entities can be defined via the `link` type. A link has one or multiple source entities and one or multiple target entities as indicted by the cardinality constraints defined for the `HasSource` and `HasTarget` associations. As mentioned earlier, we treat `Structural Links` and `Navigational Links` as two different roles that a link can have. A link has either to be a structural link or a navigational link as indicated by the partition constraint over the subcollection relationship. By modelling links as well as structural links as specialisations of the `entity` type, a structural link can have other structural links as source or target entities resulting in a multi-level composition pattern.

To support the concept of transclusions in its most general form, it is not enough that we can only define structural links over resources but we need a mechanism to address parts of a resource. In the RSL model, we therefore provide the abstract concept of a `selector` which always has to be related to a resource via the `RefersTo` association. Also for each selector type, a media-specific plug-in

has to be provided which enables the selection of parts of the related resource. A selector for XML documents could for example be an XPointer expression as explained in the previous section whereas as selector for an image might be defined by an arbitrary shape within the image.

Furthermore, arbitrary metadata in the form of key/value pairs can be added to an entity via the `HasProperties` association. The context-dependent availability of an entity can be defined by associating a set of `contextResolver` instances with an entity. Last but not least, each entity has associated information about its creator as well as information about which users might access a specific entity based on well-defined access rights. For the sake of simplicity, the user management concepts are not shown in Fig. 1. However, the full model as well as a more detailed description of the various concepts can be found in [15]. The user management component guarantees that users only see those entities for which they have the corresponding access rights. In addition, by encrypting information at entity level, privacy can be ensured when sharing documents or parts of documents.

Structures are handled by the `Structures` collection and the `HasElement` association to the corresponding structural links. It is necessary to have such an explicit grouping of structure elements since parts of structures might be reused by other structures. Furthermore, structural links are a specialisation of general links since we have to introduce an order for the substructure relationship. For example, if we want to model the structure of a document with different chapters and sections within a chapter, we have to know the order of the chapter within the entire document as well as the order of the sections within the chapters. The order over such substructure relationships is introduced via the ordered |`HasChild`| subassociation of the `HasTarget` association. Therefore, the `Structures` and |`HasChild`| components provide information about all of the components that belong to a specific structure as well as their structural relationships.

Note that since our model treats resources, selectors and links equally, structural compositions can be defined over a combination of these three entity types. Since we have a clear separation between data and structure, it is possible to reuse the same entity in different structural compositions and to support different views or structures on top of existing pieces of information as recommended by Nelson [4]. While our core model only provides some basic constructs to define navigational as well a structural links, it is up to the implementation of a particular document type to specify application-specific compositions based on these fundamental concepts. The advantage of this approach is that all document types that make use of the RSL model also offer generic access to some basic structural metadata information. This implies that we can develop general structural browsers and third-party applications can exploit the underlying structural metadata to provide additional services as explained later in Sect. 5. With our RSL model and its structural cross-media composition functionality, we were aiming for the most general form of document representation that one could imagine. While the model only offers a few basic concepts, it can be extended to

support arbitrary digital or physical document representations via the resource and selector concepts. Furthermore, our model supports navigational as well as structural linking based on these abstract media-neutral concepts.

Even if the presented solution for the structural grouping of entities is very general, it can of course also be used to implement existing solutions such as hierarchical folder structures. However, an important difference is that hierarchical ordering becomes an option and is no longer an imposition. Even if we start by organising our content in a hierarchical way, we have the flexibility to adopt the form of structural references at any later stage.

The concept of context resolvers as well as the access rights specified at entity level enable some other interesting possibilities. The structure of a document no longer has to be fixed but it can change based on some general contextual factors or based on the role of the user who is currently accessing the structural component. Similar to adaptive hypermedia, where the navigational link structure can be adapted based on a user's browsing history or some other metadata, we can adopt the structure of documents. This can, for example, be used to model highly dynamic documents that change their structure and content based on a multidimensional set of contextual parameters such as the user role or the preferred language by making use of context-dependent transclusions.

The goal of our cross-media document and link model was not to just introduce yet another hypermedia model but we wanted to find a small set of concepts that are general enough to offer some basic structural and navigational functionality for existing as well as emerging digital or physical document types.

4 Implementation

The RSL model has been implemented based on the OMS Java object database management system [17]. The resulting iServer platform offers separate Java classes for all the RSL concepts introduced in the previous section as well as a main `IServer` API that provides a set of static methods to create, access, update and delete information stored in an iServer database. The most important methods of the `IServer` API are highlighted in Fig. 2

Of course, we cannot assume that any client application that wants to make use of document metadata managed by the iServer platform has to be implemented in Java. We therefore offer an XML representation of all the data managed by iServer as well as a Web Service interface that provides the same functionality as offered by the Java API. Note that we do not primarily represent our data in XML but only use XML at the interface level to provide a language neutral interface for accessing iServer information.

In addition to the iServer core functionality, a variety of media-specific resource plug-ins have been implemented over the last few years by providing specific realisations of the corresponding resource and selector concepts. These plug-ins range from support for digital resources such a HTML pages, movie clips or Flash movies to physical resources including interactive paper or RFID tagged physical objects.

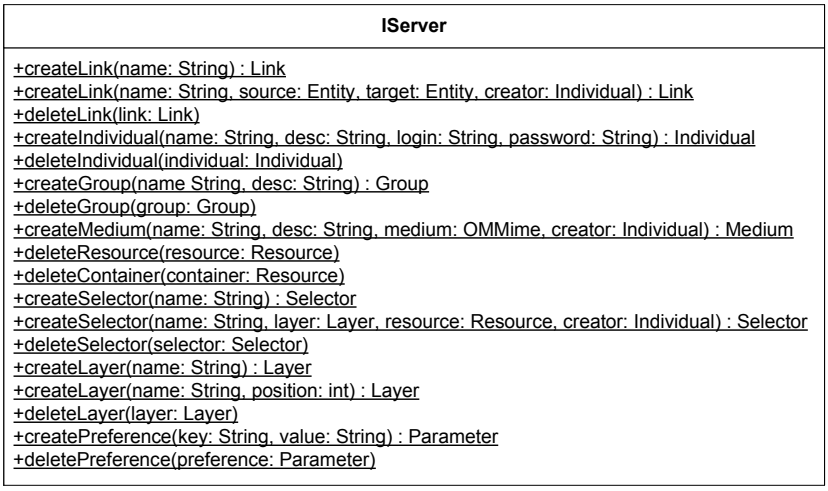

Fig. 2. Core iServer API

The implementation of a new resource plug-in includes not only the realisation of the corresponding resource and selector components to persistently store the appropriate metadata but also some visual plug-ins have to be provided. We are currently implementing a general visual cross-media authoring and browser tool that can be extended in a similar way as the data model via different visual plug-ins. The visual plug-in interface defines some common functionality to be offered by all the visual components. A media-specific visual plug-in for example has to be able to visualise a given resource or selector. Furthermore, it has to offer the functionality to create new selectors for a given resource. Based on these media-specific visualisation components, a general browser for the fluid navigation of structural cross-media documents can be developed. Note that such a general structural browser can be implemented independently of specific media types since it only has to visualise the compositional aspects between different entities whereas the visual resource plug-ins will provide the necessary rendering functionality at the resource level. Further details about our future development plans in terms of a general visual cross-media authoring and browser tool can be found in [18].

We are also investigating how the functionality of the presented RSL model can be offered at the level of a file system. There seem to be two ways how applications with their proprietary document formats can profit from such a general structural cross-media document model. They either have to be reimplemented to make use of the new API offered by an enhanced file system or plug-ins and add-ons have to be written for existing applications to enable the new structural metadata functionality. However, it could make sense to execute all applications on top of such an associative file system and at the same time offer a kind of a convenience layer that provides the old hierarchical file system view based on the new underlying structural functionality. This would enable a gradual migration

of old applications as soon as developers think that their applications could profit from the newly offered associative file system functionality.

5 Applications

In this section, we would like to evaluate the presented structural cross-media model by outlining different application scenarios and highlighting how one can profit from the existence of structural metadata. The presented structural RSL functionality is currently investigated by implementing a prototype of an RSL-based associative file system (RBAF). In addition to a file system API, the RBAF prototype offers a visual document explorer that provides access to the underlying RSL concepts to offer various metadata about a selected document. A screenshot of the initial RBAF Explorer implementation is shown in Fig. 3.

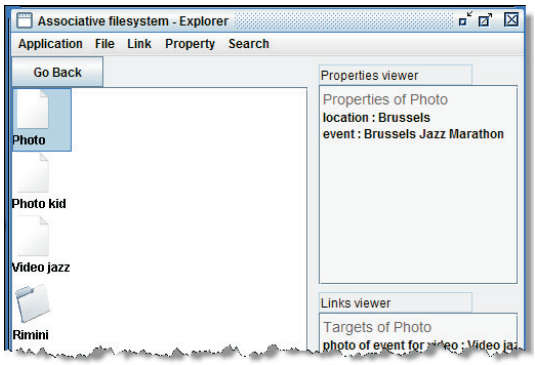

Fig. 3. RBAF Explorer

For the realisation of the RBAF file system, new file and folder resource subtypes have been introduced as shown in Fig. 4. Files can further be specialised into different types of file formats (e.g. pictures or PowerPoint presentations). Furthremore, structural links can be used to associate single files with one or more folders.

A fist application is the structural grouping of multiple documents in a project. The different documents of a project do no longer have to be copied to specific folders to ensure that they form part of the corresponding hierarchical folder structure. Often people arrange their files by media types (e.g. a folder with topical picture subfolders). However, if a picture has to be used within a specific project, it is often copied from its original picture folder to a subfolder within the corresponding project hierarchy. In our RBAF prototype this is no longer necessary, since we can just add a structural link originating from the structure defining the project to the corresponding picture. By reducing the need to copy documents just for organisational purposes, we can lower the amount

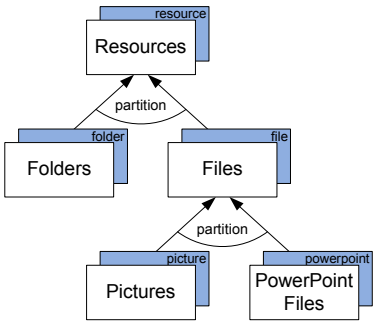

Fig. 4. RBAF model extensions

of redundant and potentially inconsistent data. Of course, the same picture can be used in multiple projects at the same time and we have still only a single copy in the original picture folder as recommended by Nelson when introducing the concept of transclusions. Even better, since all the associations in our model are bidirectional, for any picture or other document we can check whether the document is used in a project via a structural reference.

Our picture can not only be grouped with other documents, but it can also be reused as part of a document structure. In our new document representation, a PowerPoint presentation has structural links to the different slides forming part of the presentation. Each slide can further have structural links to some of its content. Our picture might therefore be used within a specific PowerPoint slide. However, we can not only use the entire image but also define a structural link to a selector addressing parts of the picture. Of course, content can also be reused on the level of single PowerPoint slides. Instead of copying specific slides from one presentation to another presentation—something that happens quite often when people want to reuse a few slides—the same slide can be reused and embedded into multiple presentations via structural links. As we have explained when introducing the context resolver concept, the structural links can also slightly vary based on contextual information and we can have slightly different versions of the same presentation without having to replicate the commonly shared slides. The same concept of variation may exist based on the current user role. While it might not be evident why we need this for PowerPoint presentations, it becomes more obvious when we talk about teaching documents in educational settings. Different structural links and thereby document versions of a word document can be provided based on user roles. For example, a student might get a word document with some questions whereas the version of the teacher contains an additional structural link to the corresponding solutions. The adaptation could even be based on individual students and their learning progress which is similar to the provision of dynamic navigational link information in adaptive hypermedia.

Note that not only applications that understand the structural within document information could profit from the additional semantics offered by the

RSL approach. Let us revisit the grouping of documents within a project. What happens if we would like to send the content of a project to a colleague? Since the corresponding project documents are no longer grouped in a single project folder, this might look like a challenge. Nevertheless, if our email client is RSL-aware, we can just select the main project file and the email client will recognise that there are structural project dependencies which means that also any structurally referenced files have to be sent to our colleague. The email client will call an RSL-aware compression tool that will automatically create a zip file containing all relevant project resources and send the resulting zip file to our colleague.

Of course the structural linking can not only be used to group different documents but it can also be applied to keep track of different versions of the same document when a document is edited. Furthermore, any structural link can be annotated with additional metadata either in terms of another link that has the structural link as a source and some other resource as target or by adding arbitrary key/value properties to the structural link.

The flexible composition of arbitrary digital or physical resources implies that some information might no be available under certain circumstances. Since we no longer follow the WYSIWYG principle, we can also no longer guarantee that all information forming part of a structural cross-media document can also be printed or visualised on another output channel. For example, if our cross-media document structure contains sound files or movie clips, then these resources cannot easily be printed on paper. On the other hand, visual components might be difficult to output via a voice output channel.

Already these small examples show plenty of possibilities how our daily work and organisation of documents could be enhanced and simplified if applications would get access to some additional structural document metadata. Of course, there are still various open issues such as whether documents have to be globally identifiable and what would be the best way to offer a general entity versioning mechanism.

6 Conclusions

We have presented our conceptual model for structural cross-media content composition and reuse and highlighted the importance of extensible document models. After reviewing some early hypermedia models and introducing the concept of transclusions, we have highlighted how existing file systems could profit from more flexible ways of organising documents and reusing parts of documents via structural references. After describing our conceptual model for extensible structural cross-media documents, we have outlined some application scenarios that can profit from having access to detailed structural document metadata. We hope that our structural cross-media model might provide a foundation for further discussions about innovative forms of document interfaces and interactions that are no longer simply simulating paper.

Acknowledgment We would like to thank Gregory Cardone for the implementation of the RBAF file system.

References

1. Bush, V.: As We May Think. Atlantic Monthly **176**(1) (1945) 101–108
2. Nelson, T.H.: Literary Machines. Mindful Press (1982)
3. Engelbart, D.C., English, W.K.: A Research Center for Augmenting Human Intellect. In: Proceedings of AFIPS Joint Computer Conferences, San Francisco, USA (December 1968) 395–410
4. Nelson, T.: Geeks Bearing Gifts: How the Computer World Got This Way. Mindful Press (2009)
5. Phelps, T.A., Wilensky, R.: The Multivalent Browser: A Platform for New Ideas. In: Proceedings of DocEng 2001, ACM Symposium on Document Engineering, Atlanta, USA (November 2001) 58–67
6. Collins, A., Apted, T., Kay, J.: Tabletop File System Access: Associative and Hierarchical Approaches. In: Proceedings of Tabletop 2007, Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems
7. Krottmaier, H., Maurer, H.: Transclusions in the 21st Century. Universal Computer Science **7**(12) (2001) 1125–1136
8. Christensen, B.G., Hansen, F.A., Bouvin, N.O.: Xspect: Bridging Open Hypermedia and XLink. In: Proceedings of WWW 2003, 12th International World Wide Web Conference, Budapest, Hungary (May 2003) 490–499
9. Koivunen, M.R.: Semantic Authoring by Tagging with Annotea Social Bookmarks and Topics. In: Proceedings of SAAW2006, 1st Semantic Authoring and Annotation Workshop, Athens, Greece (November 2006)
10. Miller, F.P., Vandome, A.F., McBrewster, J.: Office Open XML. Alphascript Publishing (2009)
11. Nürnberg, P.J., m. c. schraefel: Relationships Among Structural Computing and Other Fields. Journal of Network and Computer Applications **26**(1) (2003) 11–26
12. McGuffin, M.J., m. c. schraefel: A Comparison of Hyperstructures: Zzstructures, mSpaces, and Polyarchies. In: Proceedings of Hypertext 2004, 15th ACM Conference on Hypertext and Hypermedia, Santa Cruz, USA (August 2004) 153–162
13. Rivera, G., Norrie, M.C.: OMX-FS: An Extended File System Architecture Based on a Generic Object Model. In: Proceedings of JMLC 2000, Joint Modular Languages Conference, Zurich, Switzerland (September 2000) 161–174
14. Signer, B.: Fundamental Concepts for Interactive Paper and Cross-Media Information Spaces. Books on Demand GmbH (May 2008)
15. Signer, B., Norrie, M.C.: As We May Link: A General Metamodel for Hypermedia Systems. In: Proceedings of ER 2007, 26th International Conference on Conceptual Modeling, Auckland, New Zealand (November 2007) 359–374
16. Norrie, M.C.: An Extended Entity-Relationship Approach to Data Management in Object-Oriented Systems. In: Proceedings of ER '93, 12th International Conference on the Entity-Relationship Approach, Arlington, USA (December 1993) 390–401
17. Kobler, A., Norrie, M.C.: OMS Java: A Persistent Object Management Framework. In: Java and Databases. Hermes Penton Science (May 2002) 46–62
18. Signer, B., Norrie, M.C.: An Architecture for Open Cross-Media Annotation Services. In: Proceedings of WISE 2009, 10th International Conference on Web Information Systems Engineering, Poznan, Poland (October 2009) 387–400